Predmet: Arhitektura računara
Profesor: redovni profesor dr Dušan Regodić, dipl. inž.

Mehanizmi za efikasniji rad računara

1. *PIPELINE* MEHANIZAM
2. DMA (*DIRECT MEMORY ACCESS*) MEHANIZAM
3. MEHANIZAM PREKIDA

Pipeline mehanizam (1)

- koncept uveden sa ciljem da se ubrza proces izvršavanja programa

- zasniva se na konkurentnosti izvršavanja instrukcija Neka se proces izvršavanja instrukcije odvija u 4 faze:

1. IF (*Instruction Fatch*) → dohvatanje instrukcije iz memorije i njeno smeštanje u registar instrukcije
2. D (*Decode*) → dekodiranje koda operacije instrukcije
3. DF (*Data Fatch*) → dohvatanje operanda iz memorije i njegovo smeštanje u registar za operande
4. EX (*Execute*) → izvršavanje instrukcije

Pipeline mehanizam (1)

Pipeline predstavlja koncept koji je uveden sa ciljem da se ubrza proces izvršavanja programa. Koncept se zasniva na korišćenju konkurentnosti prilikom izvršavanja instrukcija.

- Pretpostavimo da se proces izvršavanja instrukcije odvija u četiri faze:

  IF (*Instruction Fatch*): dohvatanje instrukcije iz memorije i njeno smeštanje u registar za instrukcije

  D (*Decode*): dekodiranje operacionog koda instrukcije

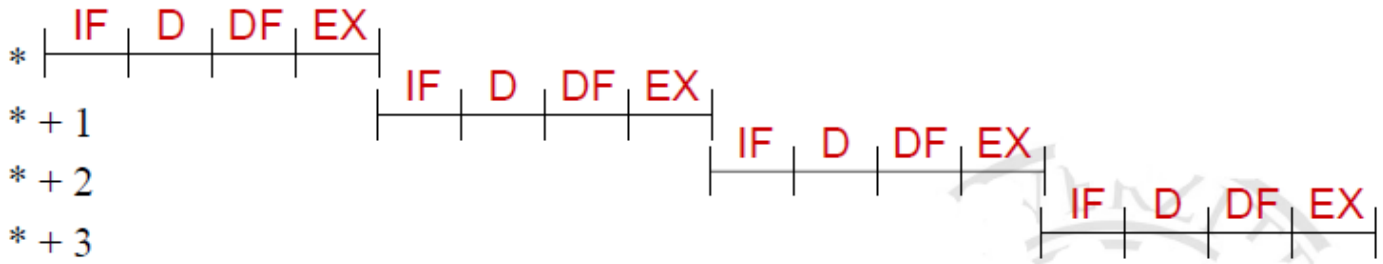
  DF (*Data Fatch*): dohvatanje podataka (operanada) bilo iz memorije ili iz odgovarajućih registara

  EX (*Execute*): izvršavanje instrukcije

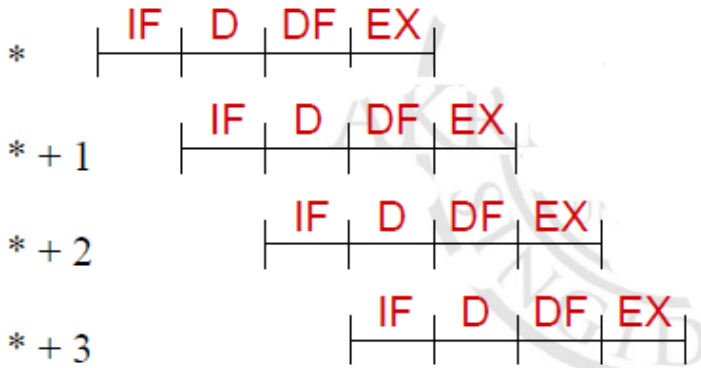
Pipeline mehanizam (2)

Primer: Izvršavanje programa od 4 instrukcije smeštene u memoriji od adrese *:

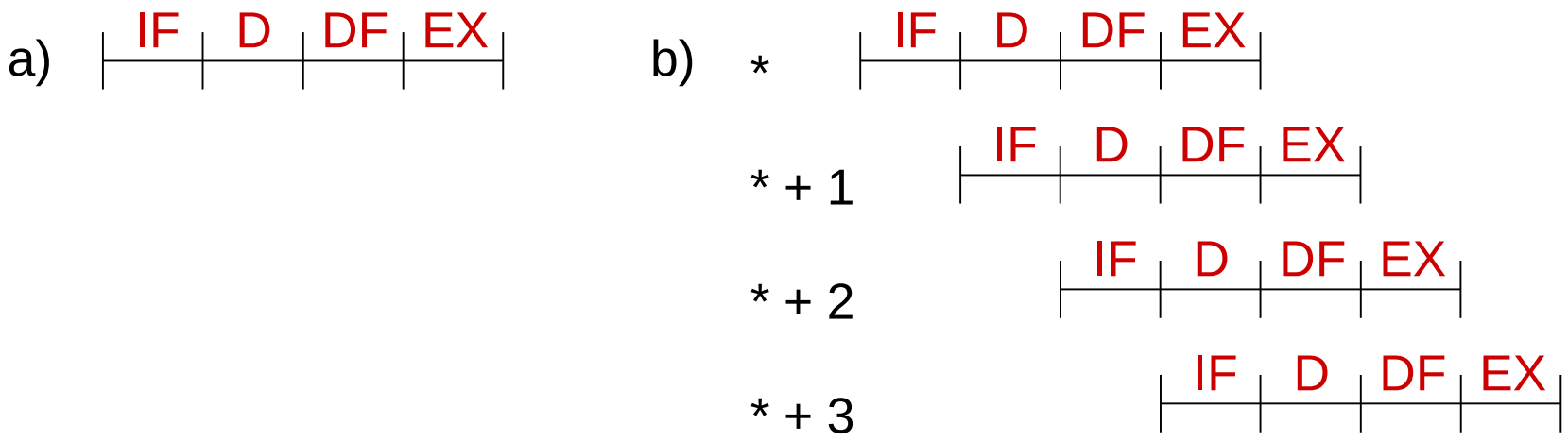
a) bez *pipeline-a*



b) sa *pipeline-om*



Pipeline mehanizam (2)



Izvršavanje instrukcija (a) obično (b) u *pipeline-u*

❑ Ubrzanje koje se može ostvariti zavisi od broja stepena (*stages*) ili dubine *pipeline*-a, tj. faza u izvršavanju instrukcije. Cena ubrzanja je u složenijem i skupljem procesoru. Ipak, skoro svi današnji procesori podržavaju *pipeline* koncept.

Pipeline mehanizam (3)

❑ **U primeru:**

- pod a) program se izvršava u 16 taktova
- pod b) program se izvršava u 7 taktova
- ostvareno ubrzanje je 56,25%

❑ubrzanje zavisi od broja stepeni (*stages*) ili dubine *pipeline*-a, tj. broja faza u izvršavanju instrukcije
❑ubrzanje se postiže po cenu složenijeg i skupljeg procesora, težeg za projektovanje
❑svi današnji procesori podržavanju *pipeline* koncept

DMA mehanizam (1)

- koncept koji omogućava direktan prenos podataka između periferije i memorije bez učešća procesora

U sistemu bez DMA:

- procesor mora da kopira deo po deo podataka i da ga prosleđuje od izvora do odredišta
- za to vreme, procesor nije raspoloživ ni za kakvu drugu aktivnost
- dodatno vreme se gubi zato što su periferije mnogo sporije od OM U sistemu sa DMA:
- procesor je oslobođen poslova oko prenosa podataka
- za vreme prenosa, procesor može da obavlja druge aktivnosti, ali ne može da koristi magistralu
- prenosom podataka upravlja posebna komponenta – DMA kontroler

Direktan pristup memoriji (1)

Direktan pristup memoriji - DMA (*Direct Memory Access)* omogućava direktan prenos podataka između periferije i memorije, bez posredovanja procesora.

- U sistemu bez DMA, procesor mora da kopira deo po deo podataka sa izvora i da ih prosleđuje do odredišta, a pri tome nije raspoloživ ni za kakvu drugu aktivnost. Dodatno vreme se gubi zato što su periferije mnogo sporije od operativne memorije.

- Uvođenjem DMA koncepta, procesor je oslobođen poslova oko prenosa podataka i za to vreme može da obavlja druge aktivnosti. Ipak, treba imati u vidu da u tim aktivnostima ne može da koristi magistralu.

Direktan pristup memoriji (1)

Direktan pristup memoriji - DMA (*Direct Memory Access)* omogućava direktan prenos podataka između periferije i memorije, bez posredovanja procesora.

- U sistemu bez DMA, procesor mora da kopira deo po deo podataka sa izvora i da ih prosleđuje do odredišta, a pri tome nije raspoloživ ni za kakvu drugu aktivnost. Dodatno vreme se gubi zato što su periferije mnogo sporije od operativne memorije.

- Uvođenjem DMA koncepta, procesor je oslobođen poslova oko prenosa podataka i za to vreme može da obavlja druge aktivnosti. Ipak, treba imati u vidu da u tim aktivnostima ne može da koristi magistralu.

Direktan pristup memoriji (2)

Princip rada

DMA prenos podatka obavlja se tako što, slanjem odgovarajuće komande DMA kontroleru, procesor inicira prenos, a zatim potpunu odgovornost preuzima DMA kontroler (generisanje adresa, slanje podataka). Po završetku prenosa, DMA kontroler obaveštava procesor da je prenos završen i da je magistrala slobodna.

- DMA prenos se primenjuje:
  - kada je potrebno preneti veliku količinu podatka, pa bi transfer preko procesora znatno usporio prenos
  - kada je periferija relativno brza (primer takve periferije je hard disk)

Mehanizam prekida (1)

Mehanizam prekida je uveden kako bi se izbeglo da procesor troši vreme čekajući na spoljašnje događaje.

Problem:
Većina periferija je znatno sporija od procesora. Ukoliko procesor treba da pošalje podatke nekoj periferiji, na primer štampaču da ih odštampa, procesor mora da sačeka da periferija završi svoj posao, tj. štampanje prispelih podataka, kako bi poslao nove podatke za štampanje. Čekanje na periferiju (dok ona obavi svoj posao) predstavlja izgubljeno vreme za procesor, jer je on tada besposlen.

Rešenje:
Navedeni problem se rešava uvođenjem mehanizma prekida koji omogućava efikasniji rad računara sa periferijama. Korišćenje prekida omogućava procesoru da izvršava druge instrukcije za vreme dok periferija obavlja svoj posao.

Mehanizam prekida (2)

- Zahvaljujući mehanizmu prekida, procesor ne čeka na oslobađanje periferije, već opslužuje periferiju na njen zahtev.

Postupak opsluživanja prekida odvija se na sledeći način:

1. Kada periferija postane spremna za prijem podataka od procesora, ona to signalizira procesoru slanjem zahteva za prekidom. Zahtev sadrži kod prekida koji odgovara periferiji koja ga je poslala.

2. Po prijemu zahteva, procesor završava izvršavanje tekuće instrukcije i na kratko prekida izvršavanje tekućeg programa kako bi opslužio prispeli zahtev.

Mehanizam prekida (3)

3. Svaki procesor ima skup prekida koje je u stanju da opslužiti. Za svaki prekid unapred je definisana tzv. prekidna rutina koja predstavlja podprogram koji treba da se izvrši u slučaju prispeća zahteva za tim prekidom. Prekidne rutine su smeštene u memoriji na određenim adresama. Sve adrese prekidnih rutina smeštene su u tabelu prekida *(Interrupt Pointer Table)* koja se, takođe, nalazi u memoriji. Procesor opslužuje prekid tako što, na osnovu koda prekida iz prispelog zahteva, pronalazi u tabeli prekida adresu odgovarajuće prekidne rutine i izvršava rutinu. U prekidnoj rutini se opslužuje odgovarajuća periferija.

4. Po završetku prekidne rutine, procesor se vraća tamo gde je stao u programu koji je izvršavao u trenutku nailaska zahteva za prekidom i nastavlja sa radom.

Mehanizam prekida (4)

- Da bi mehanizam prekida mogao uspešno da funkcioniše, neophodno je obezbediti da se pri povratku iz prekidne rutine procesor nađe i potpuno istim uslovima u kojima je bio kada je počeo opsluživanje prekida. Ti uslovi se nazivaju kontekstom procesora.

- Kad procesoru stigne zahtev za prekidom i on ga prihvati, pre nego što pređe na izvršavanje odgovarajuće prekidne rutine, procesor mora da sačuva kontekst u kome je izvršavao tekući program. Kontekst obično čine trenutni sadržaji pojedinih registara procesora.

- Kontekst je neophodno sačuvati kako bi, nakon završetka prekidne rutine, procesor znao odakle treba da nastavi izvršavanje glavnog programa i kakav je bio status nakon poslednje operacije koju je ALU izvršila pre prekida. Naime, prekidna rutina koristi iste procesorske registre kao i glavni program, pa obično menja njihov sadržaj.

Mehanizam prekida (5)

Postoje dve vrste prekida:

- spoljašnji ili eksterni prekidi - prekidi koji dolaze od periferija
- unutrašnji prekidi - prekidi koji su posledica izvršavanja instrukcije prekida, ili posledica neke neregularnosti u izvršavanju tekuće instrukcije

- Prekidima se pridružuju prioriteti koji ukazuju na njihovu važnost. Prednost u opsluživanju imaju prekidi višeg prioriteta.
- Interni prekidi su višeg prioriteta od eksternih.

Okruženje sa procesorom Intel 8086

Karakteristike procesora Intel 8086

- radni takt procesora je, u zavisnosti od verzije, 2, 5, 8, ili 10MHz
- 16-bitna magistrala podataka
- 20-bitna adresna magistrala (moguće je adresirati 2^{20} bajtova, tj. 1MB memorije)
- magistrala podataka i adresna magistrala su multipleksirane, tj. koriste iste pinove na procesorskom čipu
- postoje dve linije za spoljašnje prekide
- postoji poseban I/O (ulazno/izlazni) adresni prostor veličine 64K za adresiranje perifernih jedinica
- postoji podrška za DMA

DMA mehanizam (2)

DMA ciklus (princip rada):

- DMA kontroler obaveštava procesor da ima potrebe za DMA prenosom
- procesor završava ciklus na magistrali i šalje dozvolu DMA kontroleru da može da obavi prenos (procesor inicira prenos)
- DMA kontroler obavlja prenos (generiše adrese i signale upisa/čitanja), a za to vreme procesor može da radi nešto što ne zahteva pristup magistrali
- DMA kontroler obaveštava procesor da je prenos završen i da je magistrala slobodna

DMA mehanizam (3)

DMA prenos se primenjuje:

- kada je potrebno preneti veliku količinu podatka, pa bi transfer preko procesora znatno usporio prenos
- kada je periferija relativno brza (primer takve periferije je hard disk)

Mehanizam prekida (1)

- koncept kojim se postiže da procesor ne troši vreme čekajući na spoljašnje događaje

U sistemu bez mehanizma prekida:

- ako procesor treba da odštampa veći sadržaj, morao bi u delovima da ga šalje štampaču i da čeka da štampač odštampa prispeli deo pre nego što mu pošalje sledeći deo
- za to vreme, procesor ne bi ništa radio

U sistemu sa mehanizmom prekida:

- procesor pošalje deo podataka štampaču, a zatim obavlja svoje aktivnosti
- kada štampač završi štampanje prispelog dela, šalje zahtev za prekidom procesoru, kako bi mu ovaj poslao naredni deo sadržaja

Mehanizam prekida (2)

Procesor opslužuje periferiju na njen zahtev na sledeći način:

- periferija šalje procesoru zahtev za prekidom koji sadrži njen kôd prekida (identifikator konkretne periferije)
- po prijemu zahteva, procesor završava tekuću instrukciju, prekida izvršavanje programa i pristupa opsluživanju periferije
- svaki procesor može da opsluži određeni skup prekida; za svaki prekid definisana je prekidna rutina, tj. potprogram koji se nalazi u memoriji na određenoj adresi; u memoriji obično postoji i tabela prekida u kojoj se nalaze parovi (*kôd prekida, adresa odgovarajuće prekidne rutine*); na osnovu dobijenog kôda prekida, procesor iz tabele prekida čita adresu prekidne rutine i izvršava je; u rutini se nalaze instrukcije kojima se opslužuje periferija
- po završetku prekidne rutine, procesor se vraća na izvršavanje prekinutog programa

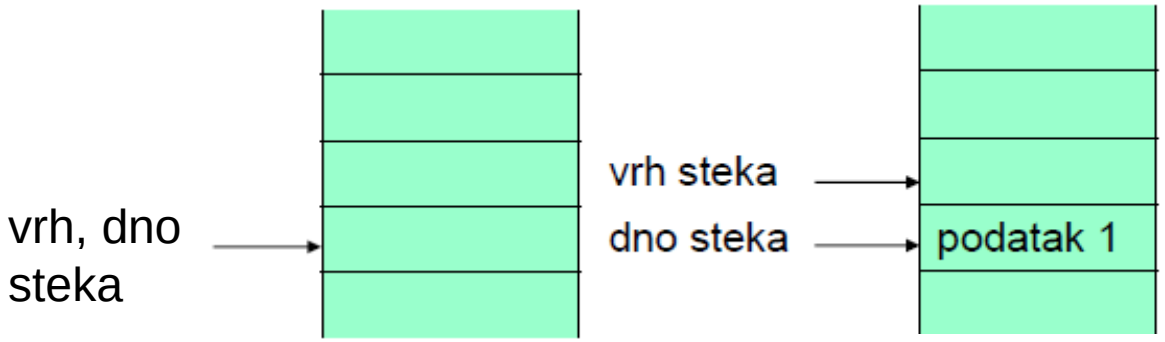
Mehanizam prekida (3)

- da bi mehanizam prekida mogao uspešno da funkcioniše, neophodno je obezbediti da se pri povratku iz prekidne rutine procesor nađe i potpuno istim uslovima u kojima je bio kada je počeo opsluživanje prekida; ti uslovi se nazivaju kontekstom procesora
- pre nego što pređe na izvršavanje odgovarajuće prekidne rutine, procesor mora da sačuva tekući kontekst (sadržaji pojedinih registara procesora)
- kontekst procesora se mora sačuvati zato što prekidna rutina koristi iste procesorske registre kao i glavni program, pa obično menja njihov sadržaj
- kontekst procesora se obično čuva na steku

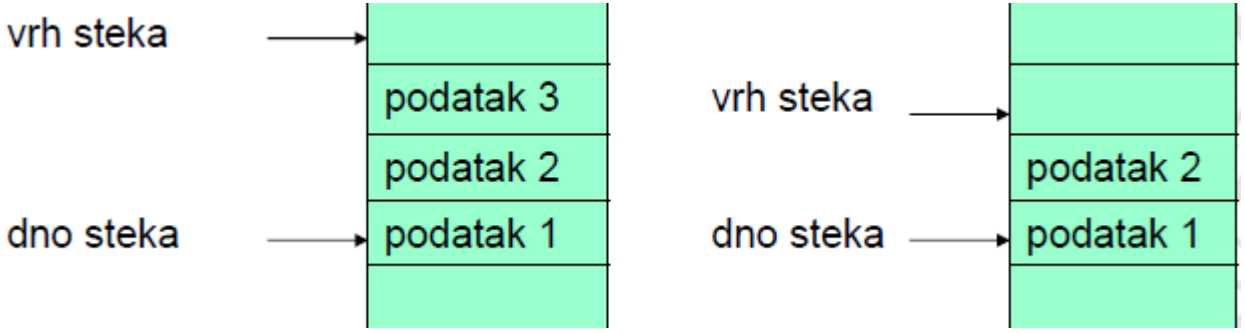
Mehanizam prekida (4)

Stek podržava LIFO (*last in, first out*) disciplinu pristupa, tj. sa steka se najpre uzima podatak koji je poslednji stavljen na stek.

Upis podatka u stek



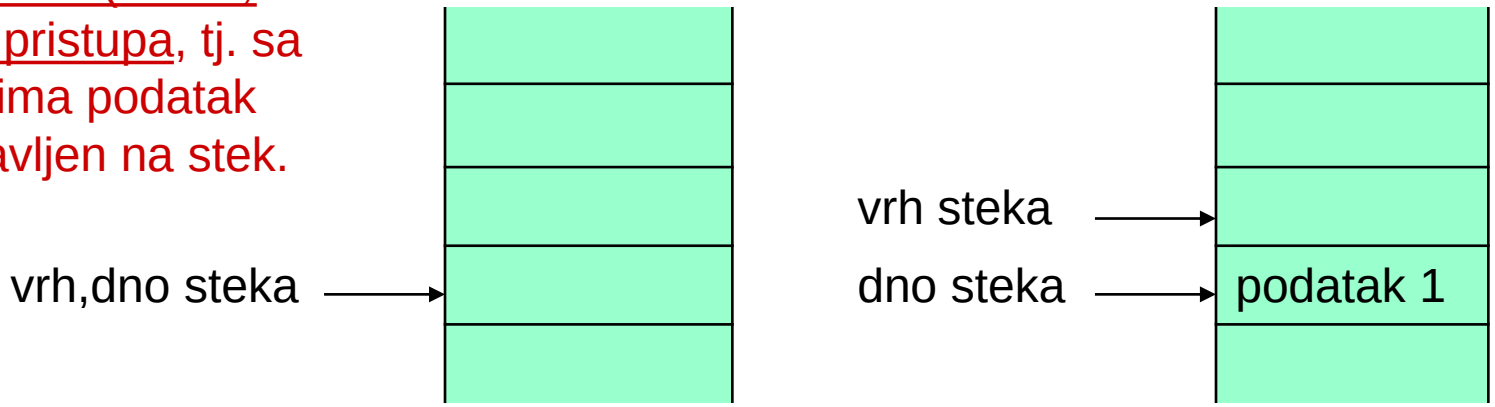
Čitanje podatka sa steka



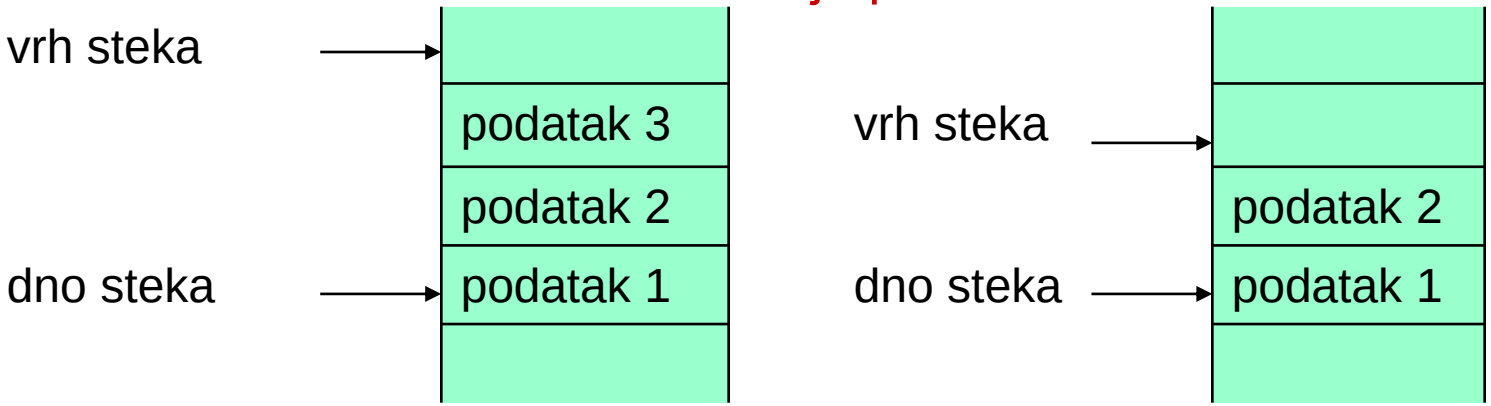
Stek

- Stek podržava LIFO (*last in, first out*) disciplinu pristupa, tj. sa steka se najpre uzima podatak koji je poslednji stavljen na stek.

Upis podatka u stek



Čitanje podatka sa steka



Mehanizam prekida (5)

Postoje dve vrste prekida:

- spoljašnji ili eksterni prekidi - prekidi koji dolaze od periferija
- unutrašnji prekidi - prekidi koji su posledica izvršavanja instrukcije prekida, ili posledica neke neregularnosti u izvršavanju tekuće instrukcije

- prekidima se pridružuju prioriteti koji ukazuju na njihovu važnost
- prednost u opsluživanju imaju prekidi višeg prioriteta
- interni prekidi su višeg prioriteta od eksternih

HVALA VAM NA PAŽNJI

Redovni profesor dr Dušan Regodić, dipl.inž.